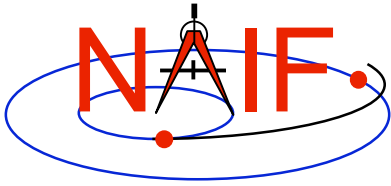


Navigation and Ancillary Information Facility

Derived Quantities

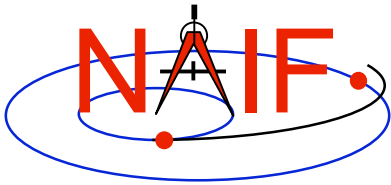
January 2009



Overview


Navigation and Ancillary Information Facility

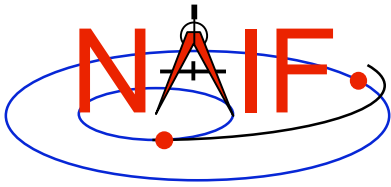
- **What are “derived quantities?”**
- **A quick tour of some of the routines provided for the computation of derived quantities**
 - Vector/Matrix Routines
 - Geometry Routines
 - Coordinate System Routines
- **Computing Illumination Angles**
- **Computing Ring Plane Intercepts**
- **Computing Occultation Events**



What are Derived Quantities?

Navigation and Ancillary Information Facility

- **Derived quantities are data produced from states, C-matrices, frame transformations, physical constants, time conversions, etc.**
 - **These are the primary reason that SPICE exists!** 
- **Examples are:**
 - Angles, Angular Rates
 - Distances, Speeds
 - Directions
 - Lighting conditions
 - Cartographic parameters
 - Schedules of events
- **The SPICE Toolkit contains many routines that assist with the computations of derived quantities.**
 - Some are fairly low level, some are quite high level.
 - More are being added as time permits.

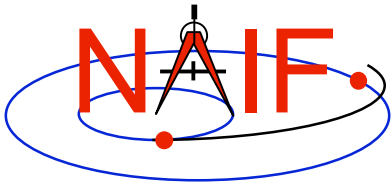


A Quick Tour

Navigation and Ancillary Information Facility

- **Vector/Matrix Routines**
 - Vector and vector derivative arithmetic
 - Matrix arithmetic
- **Geometric “Objects”**
 - Planes
 - Ellipses
 - Ellipsoids
 - Rays
- **Coordinate Systems**
 - Spherical: latitude/longitude, co-latitude/longitude, right ascension/declination; Geodetic, Cylindrical, Rectangular, Planetographic
- **Others**

The lists on the following pages are just a subset of what's available in the Toolkit.

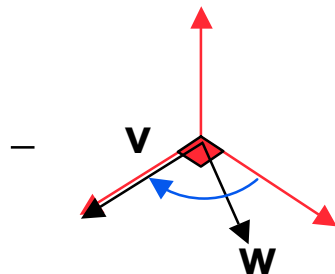
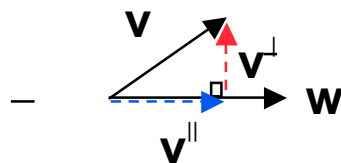


Vectors

Navigation and Ancillary Information Facility

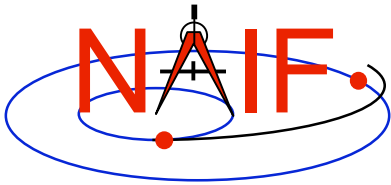
• Function

- $\langle \mathbf{v}, \mathbf{w} \rangle$
- $\mathbf{v} \times \mathbf{w}$
- $\mathbf{v}/|\mathbf{v}|$
- $\mathbf{v} \times \mathbf{w} / |\mathbf{v} \times \mathbf{w}|$
- $\mathbf{v} + \mathbf{w}$
- $\mathbf{v} - \mathbf{w}$
- $a\mathbf{v} [+ b\mathbf{w} [+ c\mathbf{u}]]$
- angle between \mathbf{v} and \mathbf{w}
- $|\mathbf{v}|$



• Routine

- $\mathbf{VDOT}, \quad \mathbf{DVDOT}$
- $\mathbf{VCROSS}, \quad \mathbf{DVCRSS}$
- $\mathbf{VHAT}, \quad \mathbf{DVHAT}$
- $\mathbf{UCROSS}, \quad \mathbf{DUCRSS}$
- $\mathbf{VADD}, \quad \mathbf{VADDG}$
- $\mathbf{VSUB}, \quad \mathbf{VSUBG}$
- $\mathbf{VSCL}, \quad [\mathbf{VLCOM}, \quad [\mathbf{VLCOM3}]]$
- \mathbf{VSEP}
- \mathbf{VNORM}
- $\mathbf{VPROJ}, \quad \mathbf{VPERP}$
- $\mathbf{TWOVEC}, \quad \mathbf{FRAME}$



Matrices

Navigation and Ancillary Information Facility

Selected Matrix-Vector Linear Algebra Routines

- Function

- $M \times v$
- $M \times M$
- $M^t \times v$
- $M^t \times M$
- $M \times M^t$
- $v^t \times M \times v$
- M^t
- M^{-1}

- Routine

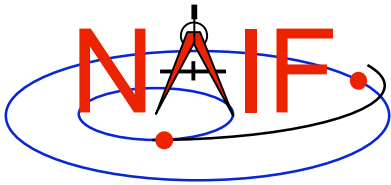
- MXV
- MXM
- MTXV
- MTXM
- MXMT
- VTMV
- XPOSE
- INVERSE, INVSTM

M = Matrix

v = Vector

x = Multiplication

T = Transpose

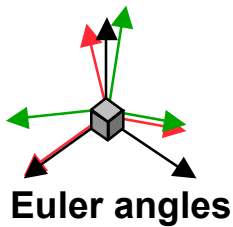


Matrix Conversions

Navigation and Ancillary Information Facility

Function

Routines

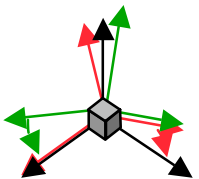


Transform between

$$\begin{matrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{matrix}$$

3x3 rotation matrix

— EUL2M, M2EUL

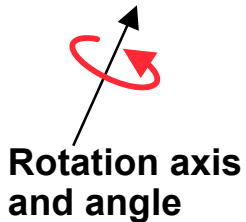


Transform between

$$\begin{matrix} a_x & a_y & a_z & & & \\ b_x & b_y & b_z & & & 0 \\ c_x & c_y & c_z & & & \\ \alpha_x & \alpha_y & \alpha_z & a_x & a_y & a_z \\ \beta_x & \beta_y & \beta_z & b_x & b_y & b_z \\ \gamma_x & \gamma_y & \gamma_z & c_x & c_y & c_z \end{matrix}$$

6x6 state transformation
matrix

— EUL2XF, XF2EUL
RAV2XF, XF2RAV



Transform between

$$\begin{matrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{matrix}$$

3x3 rotation matrix

— RAXISA, AXISAR
ROTATE, ROTMAT

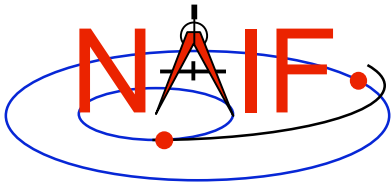
(Q_0, Q_1, Q_2, Q_3)
SPICE Style
Quaternion

Transform between

$$\begin{matrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{matrix}$$

3x3 rotation matrix

— Q2M, M2Q



Geometry

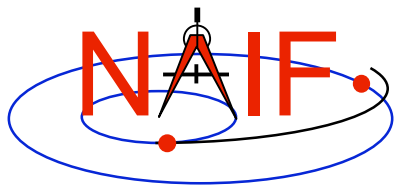
Navigation and Ancillary Information Facility

Function

- **Ellipsoids**
 - nearest point [derivative]
 - surface ray intercept
 - surface normal
 - limb
 - slice with a plane
 - altitude of ray w.r.t. to ellipsoid
- **Planes**
 - intersect ray and plane
- **Ellipses**
 - project onto a plane
 - find semi-axes of an ellipse

Routine

- NEARPT, SUBPNT, DNEARP
- SURFPT, SINCPT
- SURFNM
- EDLIMB
- INELPL
- NPEDLN
- INRYPL
- PJELPL
- SAELGV



Position Coordinate Transformations

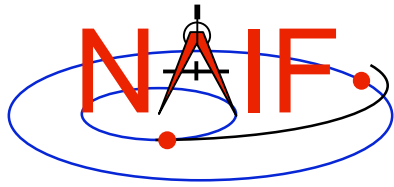
Navigation and Ancillary Information Facility

Coordinate Transformation

- Latitudinal to/from Rectangular
- Planetographic to/from Rectangular
- R.A. Dec to/from Rectangular
- Geodetic to/from Rectangular
- Cylindrical to/from Rectangular
- Spherical to/from Rectangular

Routine

- LATREC
RECLAT
- PGRREC
RECPGR
- RADREC
RECRAD
- GEOREC
REC GEO
- CYLREC
RECCYL
- SPHREC
RECS PH



Velocity Coordinate Transformations - 1

Navigation and Ancillary Information Facility

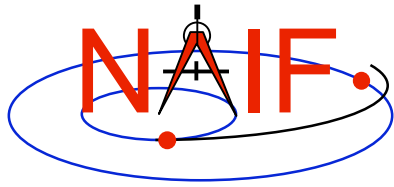
- **Coordinate Transformation**

- Latitudinal to/from Rectangular
- Planetographic to/from Rectangular
- R.A. Dec to/from Rectangular
- Geodetic to/from Rectangular
- Cylindrical to/from Rectangular
- Spherical to/from Rectangular

- **Jacobian (Derivative) Matrix Routine**

- DRDLAT
DLATDR
- DRDPGR
DPGRDR
- DRDLAT*
DLATDR*
- DRDGEO
DGEODR
- DRDCYL
DCYLDR
- DRDSPH
DSPHDR

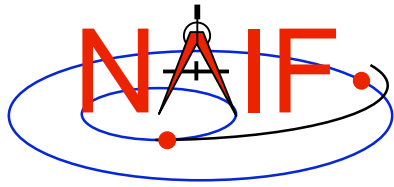
* Jacobian matrices for the R.A and Dec to/from rectangular mappings are identical to those for the latitudinal to/from rectangular mappings



Velocity Coordinate Transformations - 2

Navigation and Ancillary Information Facility

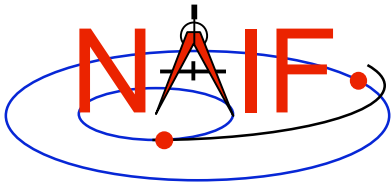
- **Transform velocities from one coordinate system to another using the SPICE Jacobian matrix routines. Example:**
 - Let (x, y, z) be a time-dependent vector expressed in rectangular coordinates:
$$(x, y, z) = \Gamma(t)$$
 - Let (α, β, γ) be the same vector expressed in spherical coordinates:
$$(\alpha, \beta, \gamma) = \Phi(\Gamma(t))$$
 - Then the chain rule gives us the time derivative of the position in spherical coordinates:
$$d(\Phi(\Gamma(t))) / dt = J(\Gamma(t)) * d\Gamma(t) / dt$$
 - The left hand side above is the velocity in spherical coordinates. The first factor on the right is the “Jacobian matrix” of the mapping from rectangular to spherical coordinates; the second factor on the right is the velocity in rectangular coordinates.



Velocity Coordinate Transformations - 3

Navigation and Ancillary Information Facility

- The SPICE calls that implement this computation are:
CALL SPKEZR (TARG, ET, REF, CORR, OBS, STATE, LT)
CALL DSPHDR (STATE(1), STATE(2), STATE(3), JACOBI)
CALL MXV (JACOBI, STATE(4), SPHVEL)
- After these calls, the vector SPHVEL contains the velocity in spherical coordinates: specifically, the derivatives
($d(r)/dt$, $d(\text{colatitude})/dt$, $d(\text{longitude})/dt$)
- Caution: coordinate transformations often have singularities, so derivatives may not exist everywhere.
 - Exceptions are described in the headers of the SPICE Jacobian matrix routines.
 - SPICE Jacobian matrix routines signal errors if asked to perform an invalid computation.

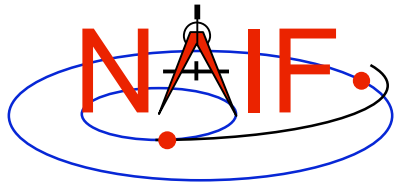


Other Derived Quantities

Navigation and Ancillary Information Facility

- **Illumination angles (phase, incidence, emission)**
 - ILUMIN*
- **Subsolar point**
 - SUBSLR*
- **Subobserver point**
 - SUBPNT*
- **Surface intercept point**
 - SINCPT*
- **Longitude of the sun (L_s), an indicator of season**
 - LSPCN

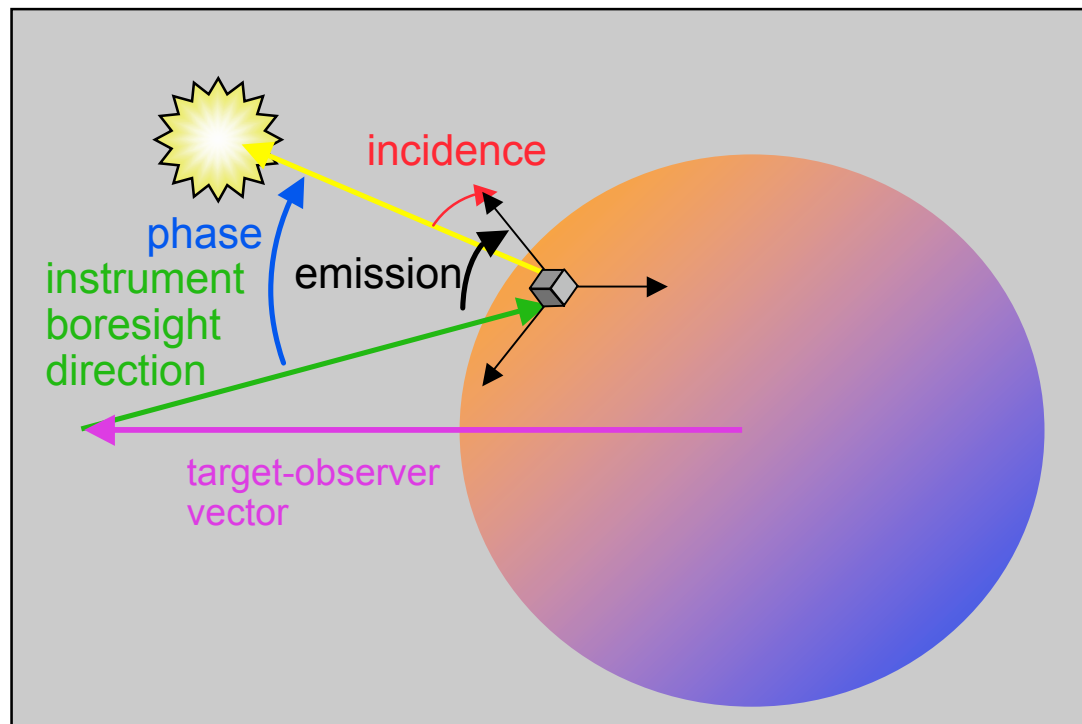
* These routines supercede the now deprecated routines ILLUM, SUBSOL, SUBPT and SRFXT

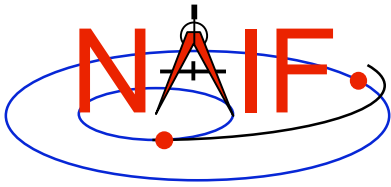


Computing Illumination Angles

Navigation and Ancillary Information Facility

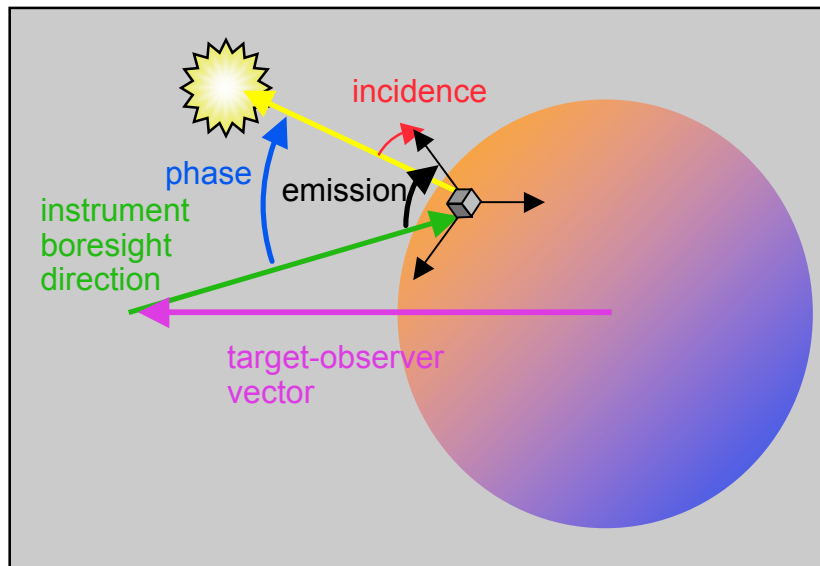
- **Given the direction of an instrument boresight in a bodyfixed frame, return the illumination angles (incidence, phase, emission) at the surface intercept on a tri-axial ellipsoid**



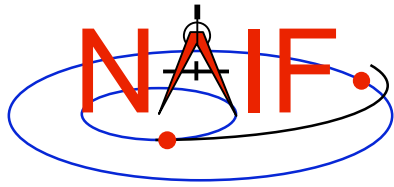


Computing Illumination Angles

Navigation and Ancillary Information Facility



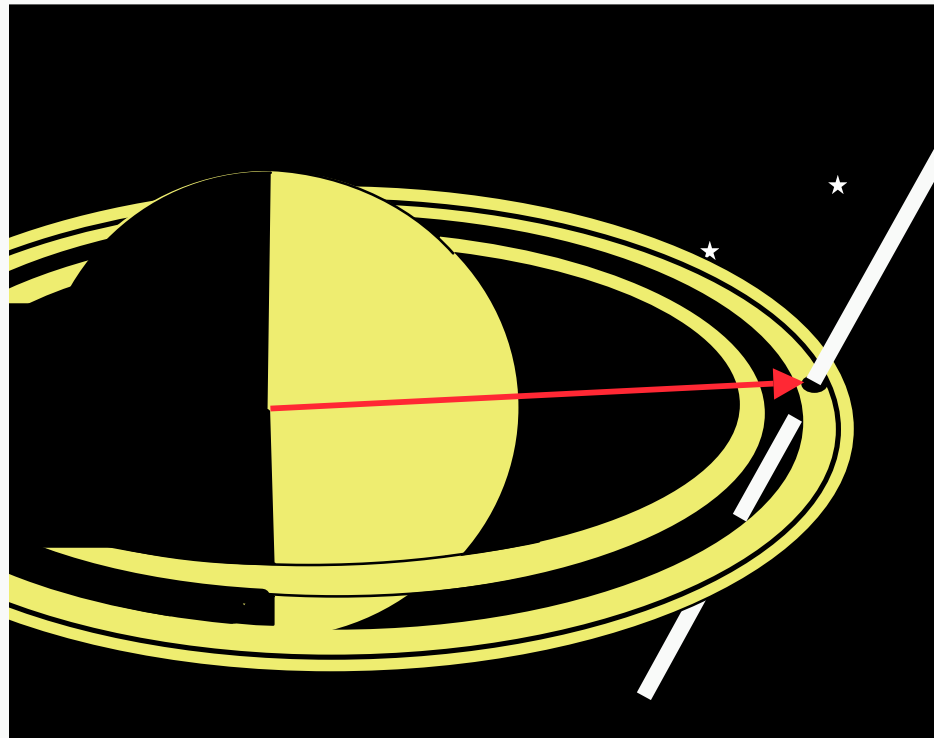
- CALL **GETFOV** to obtain boresight direction vector.
- CALL **SINCPT** to find intersection of direction vector with surface.
- CALL **ILUMIN** to determine illumination angles.

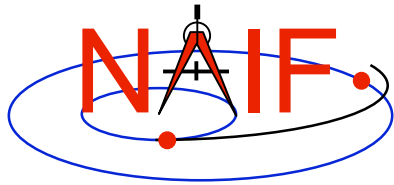


Computing Ring Plane Intercepts

Navigation and Ancillary Information Facility

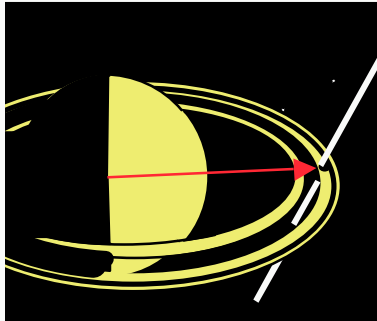
- **Determine the intersection of the apparent line of sight vector between Earth and Cassini with Saturn's ring plane and determine the distance of this point from the center of Saturn.**





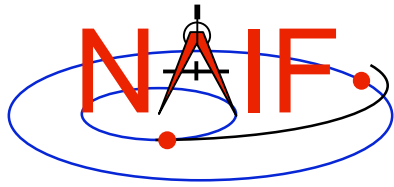
Computing Ring Plane Intercepts

Navigation and Ancillary Information Facility



This simplified computation ignores the difference between the light time from Saturn to the observer and the light time from the intercept point to the observer. It also ignores the difference in stellar aberration between the observer-Saturn vector and the observer-intercept vector.

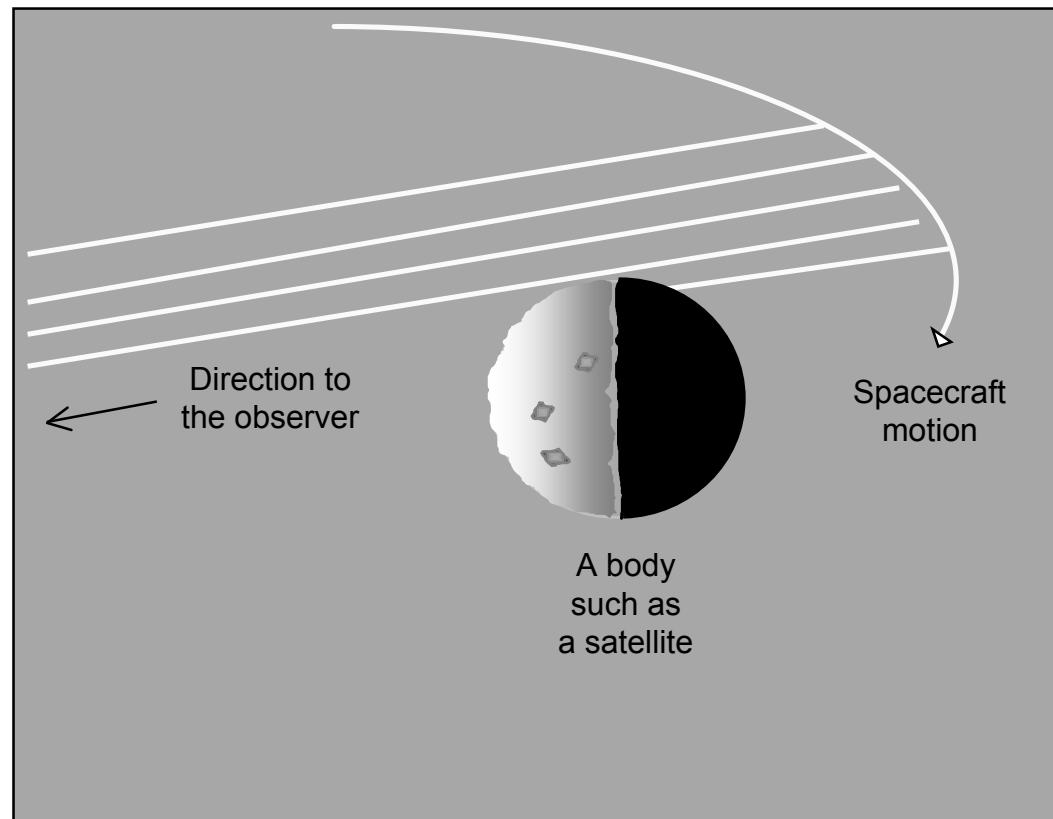
- CALL **SPKEZR** to get apparent position of spacecraft as seen from earth at time ET in J2000 reference frame SCVEC.
- CALL **SPKEZR** to get apparent position of center of Saturn at time ET as seen from earth in J2000 frame SATCTR.
- CALL **PXFORM** to get rotation from Saturn body-fixed coordinates to J2000 at light-time corrected epoch. The third column of this matrix gives the pole direction of Saturn in the J2000 frame SATPOL.
- CALL **NVP2PL** and use SATCTR and SATPOL to construct the ring plane RPLANE.
- CALL **INRYPL** to intersect Earth-spacecraft vector SCVEC with the Saturn ring plane RPLANE to produce the intercept point X.
- CALL **VSUB** to get position of intercept point with respect to Saturn XSAT (subtract SATCTR from X) and use **VNORM** to get the distance of XSAT from the center of Saturn.

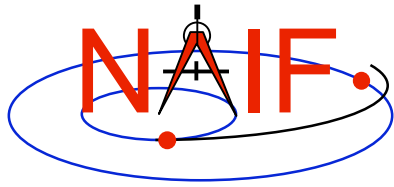


Computing Occultation Events

Navigation and Ancillary Information Facility

- Determine when the spacecraft will be occulted by an object (such as a natural satellite) as seen from an observer (such as earth).

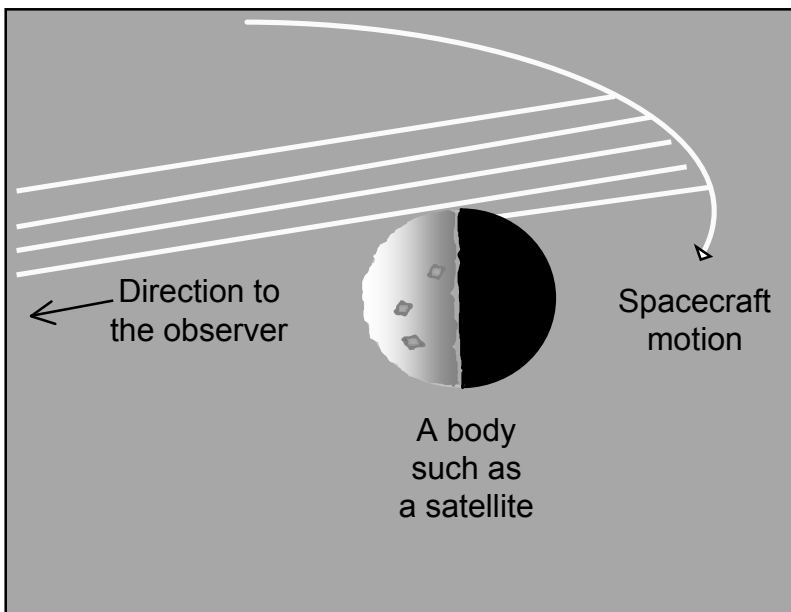


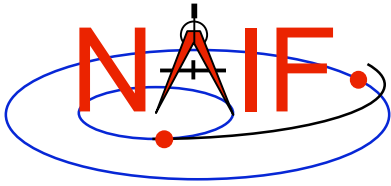


Find Epoch of Occultation Ingress

Navigation and Ancillary Information Facility

- **Select a start epoch, stop epoch and step size.**
 - Start and stop epochs should bracket a single occultation ingress epoch.
 - Step size should be smaller than occultation duration, but large enough to solve problem with reasonable speed.
- **WHILE (ET < "stop epoch")**
 - CALL **SINCPT** to see if the apparent direction of the spacecraft as seen by the observer intersects the surface of the body; do this by examining the "FOUND" flag. Take note of any such transition.
 - If an intersection is found, check that the spacecraft is further from the observer than the intersection point.
 - » Can now refine time of occultation ingress via binary search.
 - » Exit loop.
 - Otherwise, increment ET.





Event Finding Software

Navigation and Ancillary Information Facility

- **NAIF is busy building new software—the “geometry finder” subsystem—that will find a large number of geometric events, such as occultations, transits, periapsis passage, etc.**
- **A preview of these new capabilities will be provided in a later presentation.**
- **A portion of this subsystem will be provided in the N63 Toolkit... the next release.**
 - **Should be available by March 1, 2009**